

LUSTRE OVERVIEW

Nirmal Seenu
Fermilab

Lustre Features

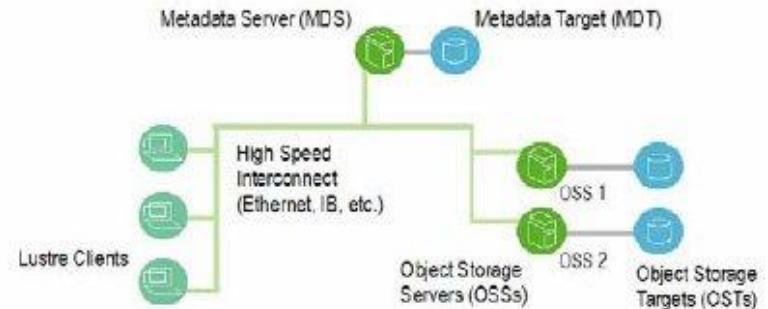
- Shared POSIX file system
- Key Characteristics
 - Aggregates many servers into one file system
 - Scales I/O throughput and capacity
 - Handles 10,000's of clients
- Built-in storage networking, including routing
- No special hardware required
 - Any block device is supported as backend storage
- Open source

Lustre Components

- Management Server(MGS)
 - The MGS stores configuration information for all Lustre file systems in a cluster
 - Each Lustre target contacts the MGS to provide information, and Lustre clients contact the MGS to retrieve information
 - The MGS requires its own disk for storage, but it could be collocated with a MDT
 - Multiple Lustre file systems can be managed by a single MGS

Lustre Components...

- Metadata Server(MDS)
 - It makes metadata available to Lustre clients via MDT
 - Each MDS manages the names and directories in the file system, and provides the network request handling for one or more local MDTs
 - It is possible to configure multiple MDTs on the same MDS but it is not recommended
- Metadata Target(MDT)
 - The MDT stores metadata (such as filenames, directories, permissions and file layout) on an MDS
 - There can be only one MDT per file system



Lustre Components...

- Object Storage Servers(OSS)
 - The OSS provides file I/O service, and network request handling for one or more local OSTs
- Object Storage Target(OST)
 - The OST stores file data
 - A single Lustre file system can have multiple OSTs, each serving a subset of file data
 - To optimize performance, a file may be spread over many OSTs. A Logical Object Volume (LOV), manages file striping across many OSTs

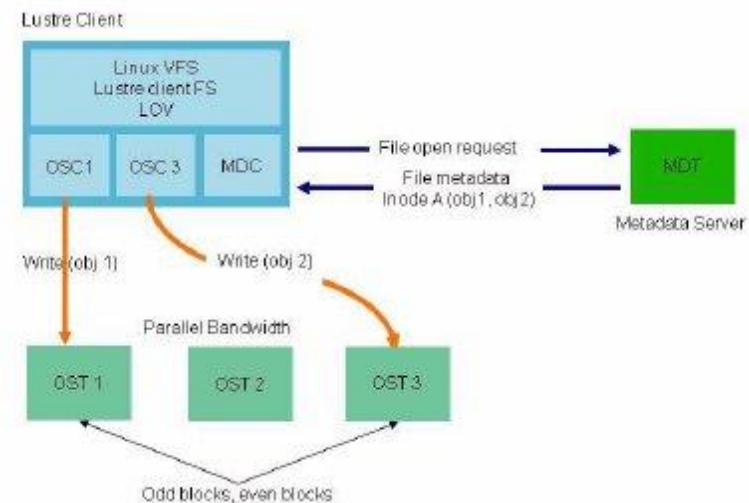
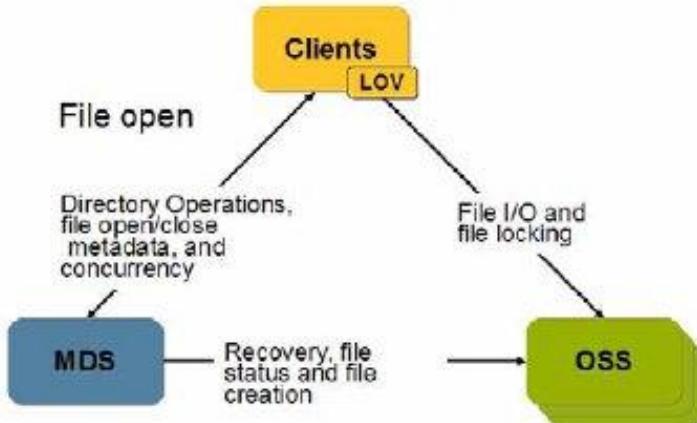
Lustre Components...

- Lustre Clients
 - The Lustre client software consists of an interface between the Linux Virtual File System and the Lustre servers
 - Each target has a client counterpart: Metadata Client (MDC), Object Storage Client (OSC), and a Management Client (MGC)
 - A group of OSCs are wrapped into a single LOV, and they provide transparent access to the file system
 - Clients which mount the Lustre file system see a single, coherent, synchronized namespace at all times
 - Different clients can write to different parts of the same file at the same time, while other clients can read from the file

Lustre Components...

- Lustre Networking (LNET)
 - It is an API that handles metadata and file I/O data for file system servers and clients
 - It supports multiple, heterogeneous interfaces on clients and servers
 - Lustre Network Drivers (LNDs) are available for a number of commodity and high-end networks, including TCP/IP, Infiniband, Myrinet (MX and GM) and Cray

File open and file I/O



Additional Lustre Features

- Interoperability
 - Lustre runs on many CPU architectures (x86, IA-64, x86-64 (EM64 and AMD64), Power PC architectures [clients only], and mixed-endian clusters; clients and servers are interoperable between these platforms)
 - Lustre supports various distributions of Linux
 - Lustre provides interoperability between adjacent software releases. Versions 1.4.x ($x > 7$), 1.6.x, and 1.8.x can interoperate with mixed clients and servers
- Access control list (ACL)
 - The Lustre security model follows a UNIX file system, enhanced with POSIX ACLs
 - It supports root squash and restriction of connections to privileged ports
- Quotas
 - User and group quotas
- OSS addition
 - The capacity of a Lustre file system and aggregate cluster bandwidth can be increased without interrupting any operations by adding a new OSS with OSTs to the cluster

Striping

- A RAID 0 pattern, in which data is "striped" across a certain number of objects
 - the number of objects is called the `stripe_count`
- Each object contains "chunks" of data
- When the "chunk" being written to a particular object exceeds the `stripe_size`, the next "chunk" of data in the file is stored on the next target



Stripping...

- Controlled striping
 - The file system has a default setting that is determined at format time
 - The default stripe count and stripe size can be tuned
 - Directories can be given an attribute so that all files under that directory (and recursively under any sub-directory) have a striping pattern determined by the attribute
 - Utilities and application libraries are provided to control the striping of an individual file at creation time
 - To change the stripe of a current file
 - Create a new directory with required stripe setting
 - Copy the file into the new directory
 - Move it back to the original location

Installing Lustre Servers

- Download the Lustre patched kernel from SUN's website. Typical installation needs the following rpms
 - kernel-lustre-smp-2.6.18-92.1.17.el5_lustre.1.6.7.1.x86_64
 - kernel-lustre-source-2.6.18-92.1.17.el5_lustre.1.6.7.1.x86_64
 - lustre-modules-1.6.7.1-2.6.18_92.1.17.el5_lustre.1.6.7.1smp.x86_64
 - lustre-ldiskfs-3.0.7.1-2.6.18_92.1.17.el5_lustre.1.6.7.1smp.x86_64
 - lustre-source-1.6.7.1-2.6.18_92.1.17.el5_lustre.1.6.7.1smp.x86_64
 - kernel-ib-1.3.1-2.6.18_92.1.17.el5_lustre.1.6.7.1smp.x86_64
 - kernel-ib-devel-1.3.1-2.6.18_92.1.17.el5_lustre.1.6.7.1smp.x86_64
 - lustre-1.6.7.1-2.6.18_92.1.17.el5_lustre.1.6.7.1smp.x86_64
- Patch the vanilla kernel.org kernel
 - Download the supported kernel.org kernel
 - Download the Lustre Source
 - Install quilt and patch the kernel as described in Lustre Manual

Formatting Lustre Servers

- Format the MGS partition
 - Its recommended that you create a small(1 GB) MGS partition
 - Multiple Lustre file systems can be managed by a single MGS
 - `mkfs.lustre --mgs --mkfsoptions="-m 0" /dev/md2`
 - `tune2fs -m 0 -i 0 -c 0 /dev/md2`
- Format the MDT partition
 - `mkfs.lustre --fsname=lqcdproj --mdt --mgsnode=ibluestre1@tcp1 --param lov.stripecount=1 --mkfsoptions="-m 0" /dev/lustre1_volume/mds_lv`
 - `tune2fs -m 0 -i 0 -c 0 /dev/lustre1_volume/mds_lv`
- Format all the OSTs
 - `mkfs.lustre --fsname=lqcdproj --mkfsoptions="-m 0" --ost --mgsnode=ibluestre1@tcp1 --param ost.quota_type=ug /dev/sdc1`
 - `tune2fs -m 0 -i 0 -c 0 /dev/sdc1`

Lustre Server and Client Options

- The default options are specified as a part of mkfs.lustre command
 - It can be tuned later with tunefs.lustre if needed
 - tune2fs can be used for ext3 tuning
- The run time options are specified in /etc/modprobe.conf
 - options Inet networks=tcp(eth1)
 - options ksocklnd tx_buffer_size=1073741824 rx_buffer_size=1073741824 enable_irq_affinity=1
- Lustre Client mount options are specified in /etc/fstab
 - lustre3@tcp:/lustre /lustre lustre defaults,flock,_netdev
- Live parameter tuning can be done using the lctl command
 - lctl conf_param lustre.sys.timeout=600
 - These changes get pushed out to all the clients and they are persistent

Installing Lustre Clients

- Client RPMs are available for certain kernels
- Patchless clients are easy to build from source
 - No reboot of worker nodes is required
 - Typical lustre client install process
 - ./configure --disable-server --enable-adaptive-timeouts --disable-snmp --with-linux=/usr/src/kernels/2.6.18-128.1.10.el5-x86_64
 - make
 - make install
 - depmod -a
 - mount -t lustre -o flock lustre3@tcp:/lustre /lustre

Hardware requirements

- Stable server hardware
 - 1 or 2 GB memory / core
 - Memory requirement depends on the number of outstanding locks
- If using GigE networks, Intel GigE cards are recommended
 - Lustre easily triggers most of the Ethernet card/driver bugs
 - Interrupt coalescing had to be enabled on Nvidia Ethernet cards

Storage Device Configuration

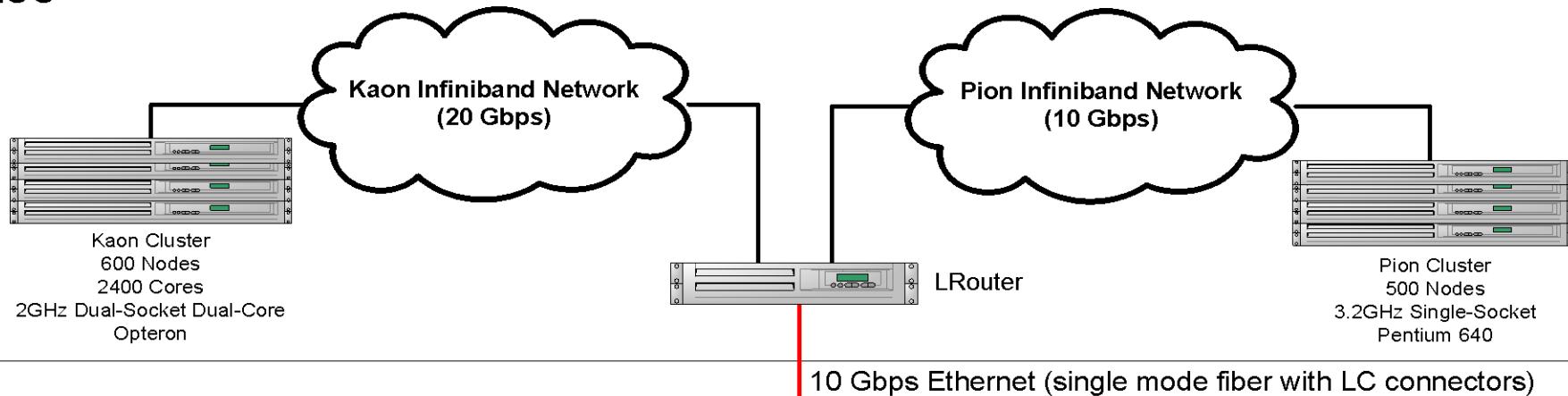
- Software RAID1 for MGS and MDT partition
 - MGS and MDT are LVM volumes on Software RAID partitions
- OST: Nexsan SATABeast
 - 42 One-TB Disks
 - 3 RAID6 volumes (12/14)
 - Four 3TB LUNS on each RAID volume
 - Battery backup on RAID controllers
 - Enable write through(disable write cache) if there is no battery backup on the RAID controller
 - SATABeast has two 4Gbps fibre channel ports
 - All LUNs are visible on both ports
- Two servers(OSS) are connected to each SATABeast
 - 6 OSTs are served by each OSS under normal operation
 - 12 OSTs can be served by one of the two OSSs on a server failure

LQCD Lustre Server Configuration

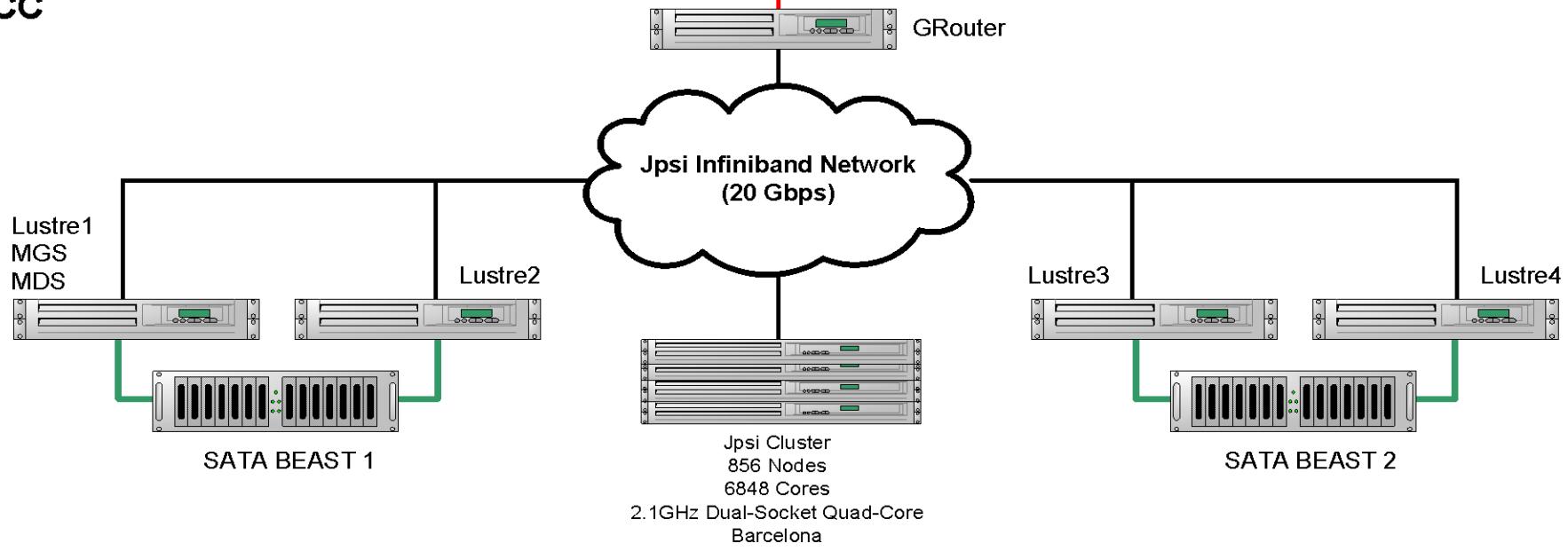
- Single socket Intel Xeon Quad Core
 - E5420 @ 2.50GHz
 - 4 GB RAM
 - Supermicro X7DBU motherboards
 - Intel 80003ES2LAN Gigabit Ethernet Controller
 - Mellanox MT25204 [InfiniHost III Lx HCA]
 - IPoIB and native IB are used
 - QLogic QLE2460 4Gb Fibre Channel

LQCD Lustre Server Configuration

LCC



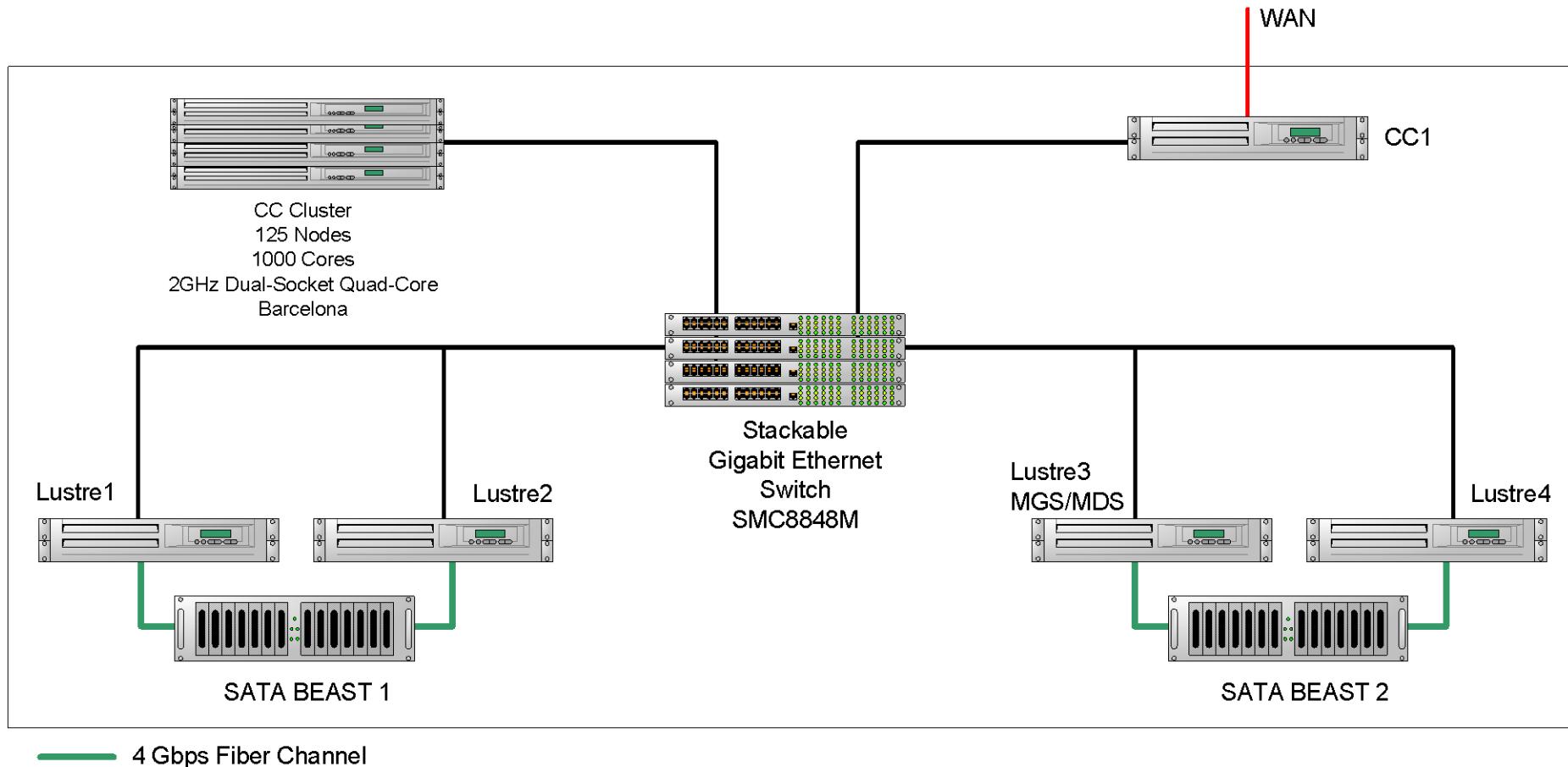
GCC



CC Lustre Server Configuration

- Dual socket AMD Barcelona Quad Core
 - AMD Opteron 2350 @ 2 GHz
 - 16 GB RAM
 - Supermicro H8DMT motherboards
 - nVidia MCP55 Ethernet
 - QLogic QLE2460 4Gb Fibre Channel

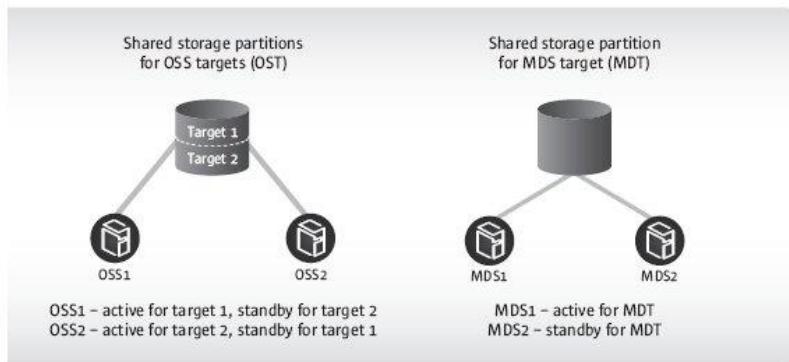
CC Clustre Lustre Architecture



Backup

- Metadata backup
 - Our Metadata servers are 40GB(unformatted) in size
 - The unused portions of the disk was 'zero'ed out to enable better compression
 - dd if=/dev/zero of=/mnt/mdt/tmp/gbfile bs=128k count=8192
 - Create the LVM Snapshots
 - /usr/sbin/lvcreate -L40G -s -n mdtsnapshot /dev/lustre1_volume/mds_lv
 - Create the Backup Image for MDT
 - /bin/dd if=/dev/lustre1_volume/mdtsnapshot | gzip -c > /mnt/mdtbackup/mdt-backup.img.gz
 - Remove the Snapshot
 - /usr/sbin/lvremove -f /dev/lustre1_volume/mdtsnapshot >> \$LOGFILE
2>&1
- User data Backup
 - Its hard to take a valid/consistent snapshot across all OSTs
 - Done from a lustre client using regular backup tools

High Availability and Rolling Upgrades



- This failover mechanism, in conjunction with software that offers interoperability between versions, is used to support rolling upgrades of file system software on active clusters
- Lustre MDSs are configured as an active/passive pair
- OSSs are typically deployed in an active/active configuration that provides redundancy without extra overhead

Scalability and Performance

- LQCD - Lustre file system has ~2000 clients running with IPoIB without any problem
 - Mounting ~2000 clients takes a few seconds
 - Un-mounting clients takes a little longer depending on the usage pattern on the clients
 - Performance is limited by the amount of data that can be transferred in and out of SATABeast
- CC – Lustre file system
 - Performance is limited by the GigE network

Data Safety and Replication

- The client-OST connection guarantees data integrity
- Clients caches transactions and replays failed transactions if needed
- Client cache is released on write acknowledgement from the MDS
- No support for data replication. It might be available in the future versions

Monitoring and Admin tools

- Statistics and health information is available in the /proc file system on Lustre Server and Clients:
 - /proc/sys/lustre
 - /proc/fs/lustre
- “lfs check servers” list the status of all servers
- Checking for the presence of a file on the lustre client seems to be sufficient in most cases.
- Periodic checking for “Call Trace” in the dmesg output.

Bestman Installation

- Lqcsrm is the Bestman/GridFTP server
 - This hardware is the same as the LQCD Lustre servers
 - lustre client is mounted on this node at /lqcdproj
- Use pacman to install Bestman
 - pacman -get http://vdt.cs.wisc.edu/vdt_11098_cache:Bestman
- Configure the certificate update script
- Configure PRIMA
- Configure Bestman
 - `./vdt/setup/configure_bestman --server y --user daemon --cert /etc/grid-security/osg-se/hostcert.pem --key /etc/grid-security/osg-se/hostkey.pem --gums-host gums.fnal.gov --gums-port 8443 --gums-dn /DC=org/DC=doegrids/OU=Services/CN=lqcdrsm.fnal.gov --enable-gateway --with-transfer-servers gsiftp://lqcdrsm.fnal.gov --with-allowed-paths=/lqcdproj;/scratch --globus-tcp-port-range 40000,41000 --http-port 8080 --https-port 8443 -cksm_type=null`

Backup Slides

/proc/sys/lustre

```
[root@lustre3 lustre]# ls -l
```

```
-rw-r--r-- 1 root root 0 Jun 12 10:29 alloc_fail_rate
-rw-r--r-- 1 root root 0 Jun 12 10:29 debug_peer_on_timeout
-rw-r--r-- 1 root root 0 Jun 12 10:29 dump_on_eviction
-rw-r--r-- 1 root root 0 Jun 12 10:29 dump_on_timeout
-rw-r--r-- 1 root root 0 Jun 12 10:29 fail_loc
-rw-r--r-- 1 root root 0 Jun 12 10:29 fail_val
-rw-r--r-- 1 root root 0 Jun 12 10:29 ldlm_timeout
-rw-r--r-- 1 root root 0 Jun 12 10:29 max_dirty_mb
-r--r--r-- 1 root root 0 Jun 12 10:29 memused
-r--r--r-- 1 root root 0 Jun 12 10:29 memused_max
-r--r--r-- 1 root root 0 Jun 12 10:29 pagesused
-r--r--r-- 1 root root 0 Jun 12 10:29 pagesused_max
-rw-r--r-- 1 root root 0 Jun 12 10:29 timeout
```

/proc/fs/lustre

```
[root@lustre3 lustre]# ls -l
```

```
-r--r--r-- 1 root root 0 Jun 12 10:28 devices
-r--r--r-- 1 root root 0 Jun 12 10:28 health_check
dr-xr-xr-x 4 root root 0 Jun 12 10:28 ldlm
dr-xr-xr-x 2 root root 0 Jun 12 10:28 llite
dr-xr-xr-x 3 root root 0 Jun 12 10:28 lov
dr-xr-xr-x 9 root root 0 Jun 12 10:28 lquota
dr-xr-xr-x 2 root root 0 Jun 12 10:28 mdc
dr-xr-xr-x 3 root root 0 Jun 12 10:28 mds
dr-xr-xr-x 3 root root 0 Jun 12 10:28 mdt
dr-xr-xr-x 3 root root 0 Jun 12 10:28 mgc
dr-xr-xr-x 3 root root 0 Jun 12 10:28 mgs
dr-xr-xr-x 8 root root 0 Jun 12 10:28 obdfilter
dr-xr-xr-x 26 root root 0 Jun 12 10:28 osc
dr-xr-xr-x 3 root root 0 Jun 12 10:28 ost
-r--r--r-- 1 root root 0 Jun 12 10:28 pinger
-r--r--r-- 1 root root 0 Jun 12 10:28 version
```

/proc/fs/lustre/obdfilter/lustre-OST0012

```
[root@lustre3 lustre-OST0012]# ls -l
```

```
-r--r--r-- 1 root root 0 Jun 12 10:38 blocksize
-r--r--r-- 1 root root 0 Jun 12 10:38 brw_stats
-rw-r--r-- 1 root root 0 Jun 12 10:38 client_cache_count
-rw-r--r-- 1 root root 0 Jun 12 10:38 client_cache_seconds
--w----- 1 root root 0 Jun 12 10:38 evict_client
dr-xr-xr-x 129 root root 0 Jun 12 10:38 exports
-r--r--r-- 1 root root 0 Jun 12 10:38 filegroups
-r--r--r-- 1 root root 0 Jun 12 10:38 filesfree
-r--r--r-- 1 root root 0 Jun 12 10:38 filestotal
-r--r--r-- 1 root root 0 Jun 12 10:38 fstype
-r--r--r-- 1 root root 0 Jun 12 10:38 hash_stats
-r--r--r-- 1 root root 0 Jun 12 10:38 kbytesavail
-r--r--r-- 1 root root 0 Jun 12 10:38 kbytesfree
-r--r--r-- 1 root root 0 Jun 12 10:38 kbytestotal
-r--r--r-- 1 root root 0 Jun 12 10:38 last_id
-r--r--r-- 1 root root 0 Jun 12 10:38 mntdev
-r--r--r-- 1 root root 0 Jun 12 10:38 num_exports
-rw-r--r-- 1 root root 0 Jun 12 10:38 quota_btune_sz
-rw-r--r-- 1 root root 0 Jun 12 10:38 quota_bunit_sz
-rw-r--r-- 1 root root 0 Jun 12 10:38 quota_itune_sz
-rw-r--r-- 1 root root 0 Jun 12 10:38 quota_iunit_sz
-rw-r--r-- 1 root root 0 Jun 12 10:38 quota_switch_seconds
-rw-r--r-- 1 root root 0 Jun 12 10:38 quota_type
-rw-r--r-- 1 root root 0 Jun 12 10:38 readcache_max_filesize
-r--r--r-- 1 root root 0 Jun 12 10:38 recovery_status
-rw-r--r-- 1 root root 0 Jun 12 10:38 stats
-r--r--r-- 1 root root 0 Jun 12 10:38 tot_dirty
-r--r--r-- 1 root root 0 Jun 12 10:38 tot_granted
-r--r--r-- 1 root root 0 Jun 12 10:38 tot_pending
-r--r--r-- 1 root root 0 Jun 12 10:38 uid
```

/proc/fs/lustre/obdfilter/lustre-OST0012/brw-stats

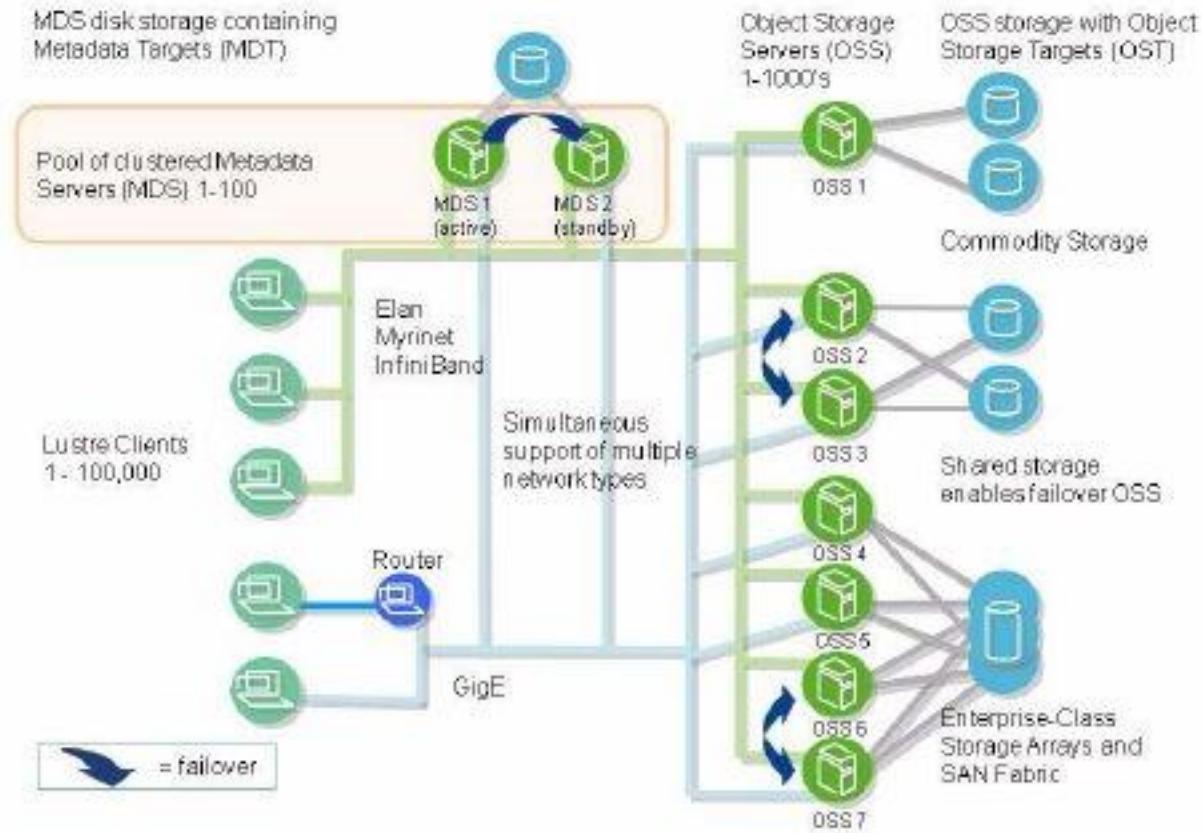
snapshot_time: 1244821207.889622 (secs.usecs)

		read				write							
pages per bulk	r/w	rpcs	% cum %		rpcs	% cum %	I/O time (1/1000s)	ios	% cum %		ios	% cum %	
1:		187354	32 32		40670	6 6	1:	143005	24 24		74850	12 12	
2:		1386	0 32		6336	1 7	2:	1301	0 24		51740	8 20	
4:		1889	0 32		2122	0 8	4:	40928	7 31		265027	43 64	
8:		3739	0 33		2141	0 8	8:	176144	30 62		130869	21 85	
16:		5429	0 34		5452	0 9	16:	71121	12 74		56747	9 95	
32:		5688	0 35		10761	1 11	32:	82984	14 88		21602	3 98	
64:		7537	1 36		23962	3 14	64:	52322	8 97		4906	0 99	
128:		9117	1 38		41612	6 21	128:	12021	2 99		2180	0 99	
256:		359654	61 100		476572	78 100	256:	1628	0 99		1008	0 99	
		read				write		512:	285	0 99		587	0 99
		disk I/O size	ios	% cum %		ios	% cum %	1K:	47	0 99		85	0 99
4K:		53222	11 11		44097	5 5	2K:	7	0 100		20	0 99	
8K:		1962	0 12		11202	1 7	4K:	0	0 100		7	0 100	
16K:		2080	0 12		12409	1 8							
32K:		3842	0 13		6593	0 9							
64K:		5571	1 14		16624	2 11							
128K:		5943	1 16		29349	3 15							
256K:		7897	1 17		132621	16 32							
512K:		9562	2 20		113189	14 46							
1M:		359423	79 100		423307	53 100							

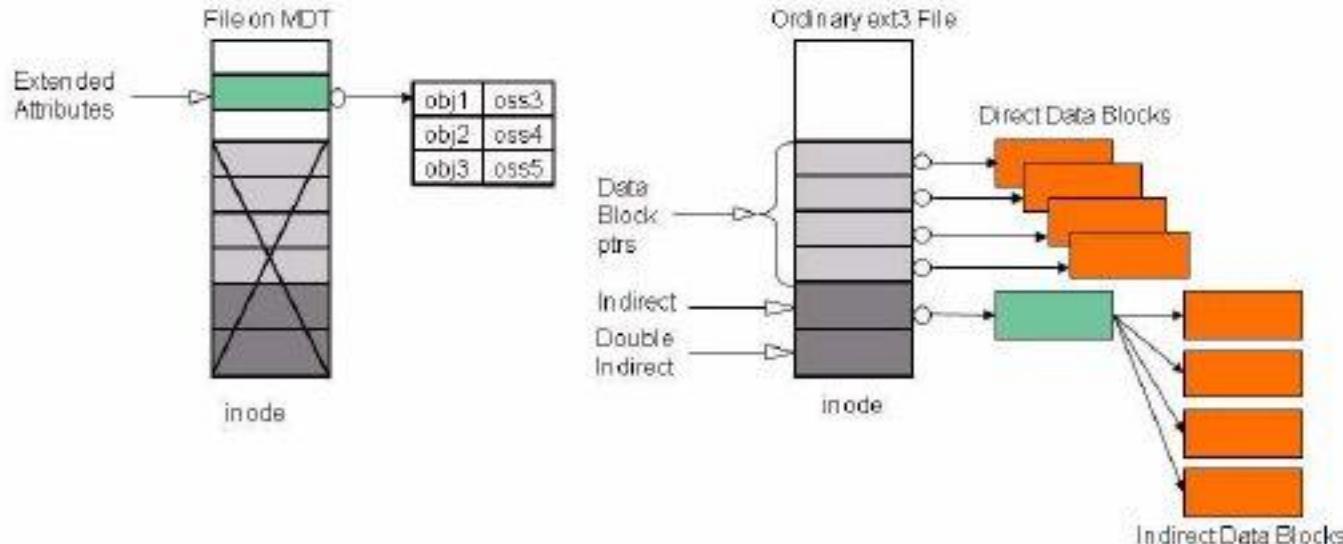
Upgrade Experiences...

- A clean shutdown of all Lustre servers and clients is preferred on upgrades
 - Automatic recovery process take a long time if clients weren't shutdown cleanly
 - Recovery process can be cancelled
 - All the cached transactions on the client side are cancelled

Lustre Architecture



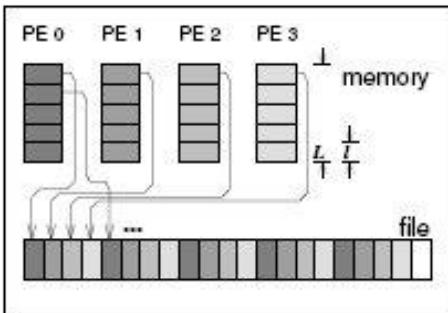
File on MDT



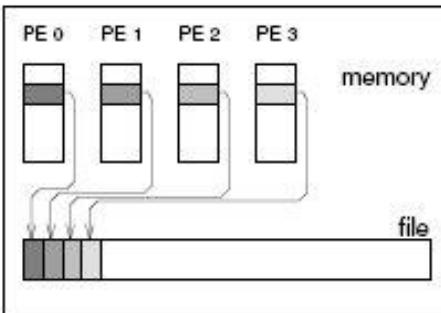
MDS inodes point to objects, ext3 inodes point to data

B_eff_io Pattern Details

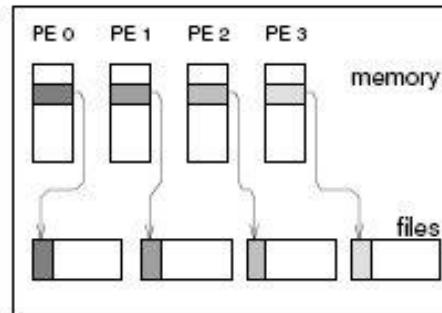
- Access methods:
 - initial write
 - Rewrite
 - Read
- Transfer Patterns
 - (0) striped collective access, scattering large chunks in memory with size “ L ” each with one MPI-I/O call to/from disk chunks with size “ l ”
 - (1) striped collective access, but one read or write call per disk chunk
 - (2) noncollective access to one file per MPI process, i.e., on separated files
 - (3) same as (2), but the individual files are assembled to one segmented file
 - (4) same as (3), but the access to the segmented file is done with collective routines



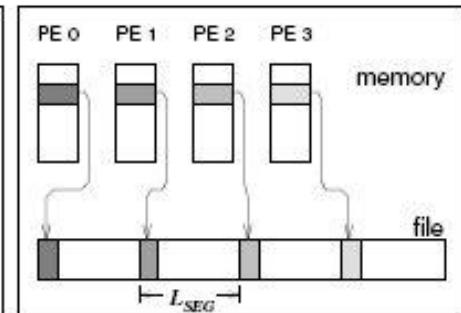
pattern type 0



pattern type 1



pattern type 2



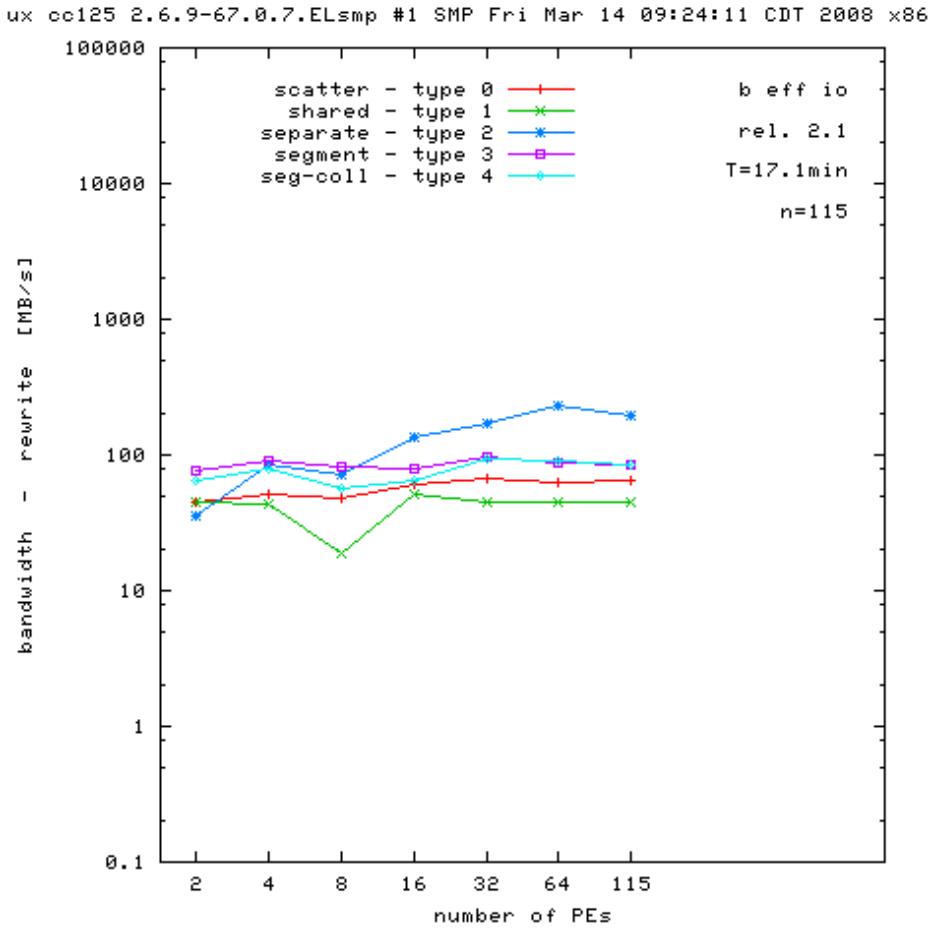
pattern type 3/4

Pattern Sizes

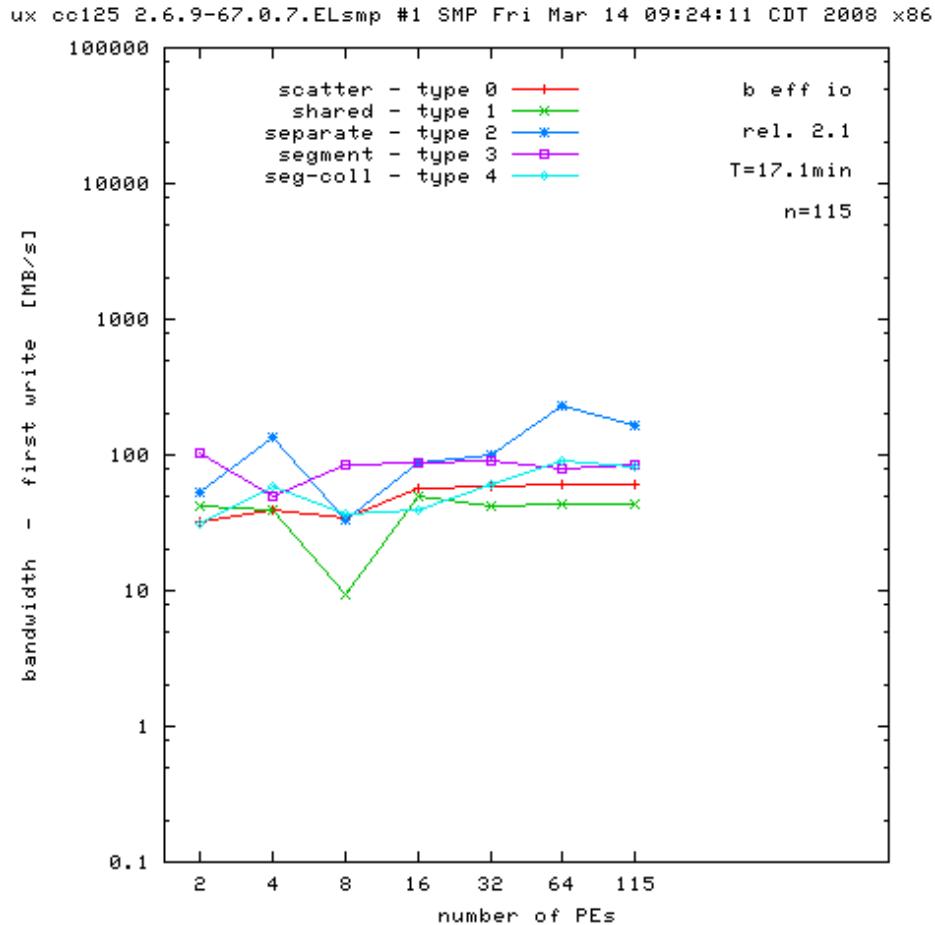
Pattern Type	No.	<i>l</i>	<i>L</i>	<i>U</i>
0: scatter, collect.	0	1 MB	1 MB	0
	1	M_{PART}	$:=l$	4
	2	1 MB	2 MB	4
	3	1 MB	1 MB	4
	4	32 kB	1 MB	2
	5	1 kB	1 MB	2
	6	32 kB +8B	1 MB + 256B	2
	7	1 kB +8B	1 MB + 8kB	2
	8	1 MB +8B	1 MB + 8B	2
1: shared, collect.	9	1 MB	$:=l$	0
	10	M_{PART}	$:=l$	4
	11	1 MB	$:=l$	2
	12	32 kB	$:=l$	1
	13	1 kB	$:=l$	1
	14	32 kB +8B	$:=l$	1
	15	1 kB +8B	$:=l$	1
	16	1 MB +8B	$:=l$	2

Pattern Type	No.	<i>l</i>	<i>L</i>	<i>U</i>
2: separated files, non-coll.	17	1 MB	$:=l$	0
	18	M_{PART}	$:=l$	2
	19	1 MB	$:=l$	2
	20	32 kB	$:=l$	1
	21	1 kB	$:=l$	1
	22	32 kB +8B	$:=l$	1
	23	1 kB +8B	$:=l$	1
	24	1 MB +8B	$:=l$	2
3: segmented, non-coll.	25f	same as patterns 17–24		
	33	fill up segments	$:=l$	0
4: segmented, collective	34f	same as patterns 25–33		
$\Sigma U = 64$				

CC Lustre Effective I/O Bandwidth benchmark



CC Lustre b_eff_io ...



CC Lustre b_eff_io...

